# Chapter 13 VB.net 2008 Program Running in WP-8xx8 Access to Win-GRAF Variables

This chapter lists the procedure for creating the first demo program by Visual Studio .NET 2008 development tool. There are some sample programs in the WP-8xx8 CD-ROM.

**VB .NET example:**

CD-ROM : \napdos\Win-GRAF\demo-project\vb.net_2008_demo\

demo_vb01 : Digital I/O demo with one I-87055W in slot 0 of the WP-8xx8.

demo_vb02 : Analog I/O demo with one I-87024W in slot 1, one I-8017HW in slot 2.

demo_vb03 : Read/Write Win-GRAF internal integers, timers & real variables. (No I/O)

demo_vb04 : Read/Write Win-GRAF internal String variables. (No I/O)

**Win-GRAF example:**

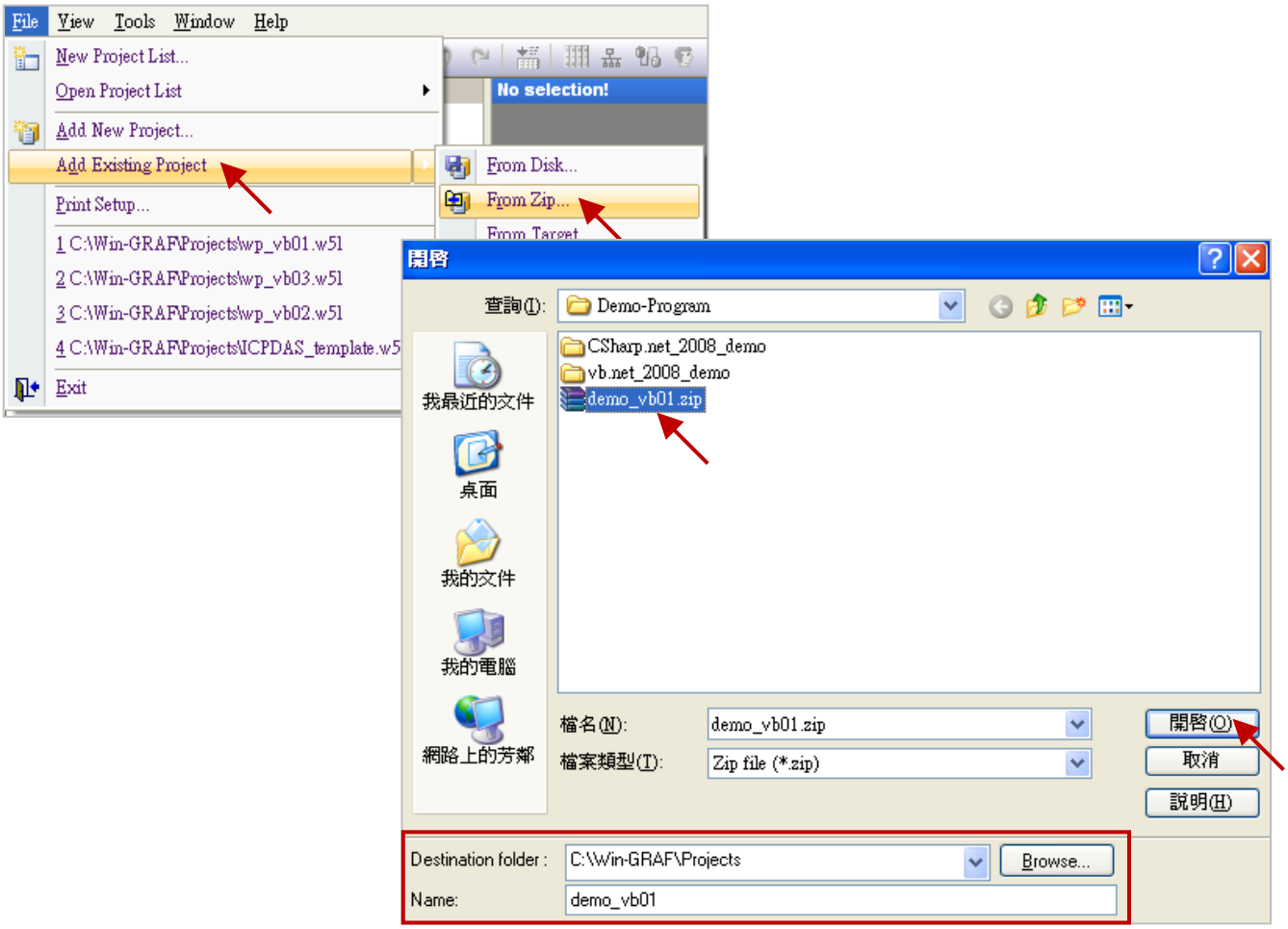
CD-ROM : \napdos\Win-GRAF\demo-project\

"demo_vb01.zip", "demo_vb02.zip", "demo_vb03.zip", "demo_vb04.zip"

## 13.1 Add an Existing Win-GRAF Project from a ZIP

Please follow these steps to restore the Win-GRAF project.

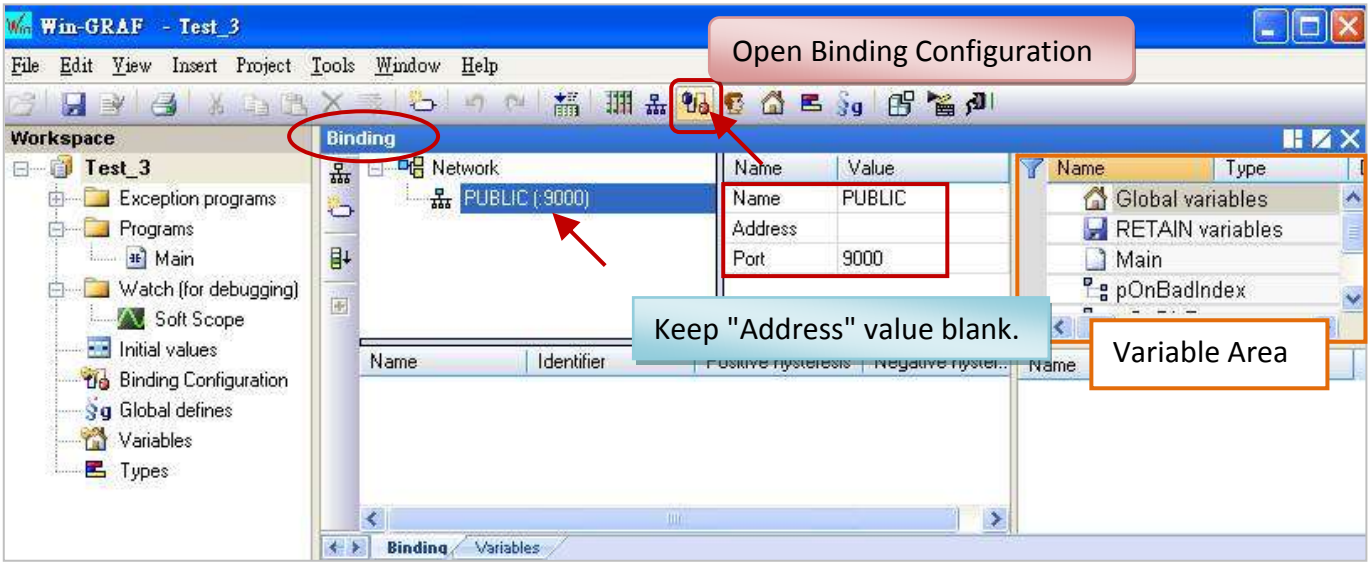
First Click “File” -> “Add Existing Project” -> “FromZip…”. Then choose the Win-GRAF project zip file which you would like to restore. After restoring the project, you have to build the project, and then download it to the PAC.



## 13.2 Publishing the Win-GRAF Variable for .NET and Soft-GRAF HMI

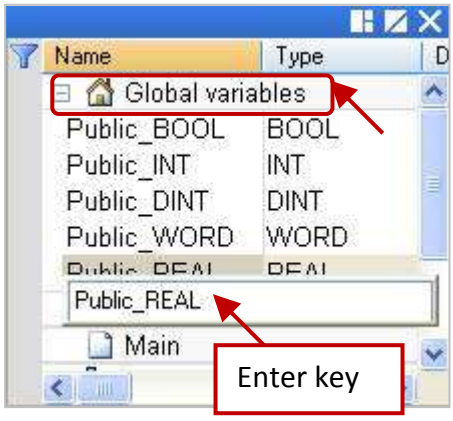
If users wish to use .NET program to Read/Write the Win-GRAF variables. Except for String variable all of the variables need to use the "Open Binding Configuration" function to set an address. The following demonstrates how to publish Win-GRAF variable:

1. Click “Open Binding Configuration” on the toolbar to open the “Binding” setup window.
2. Click "PUBLIC (:9000)". Keep the "Address" value is blank and "Port" value is fixed to 9000.



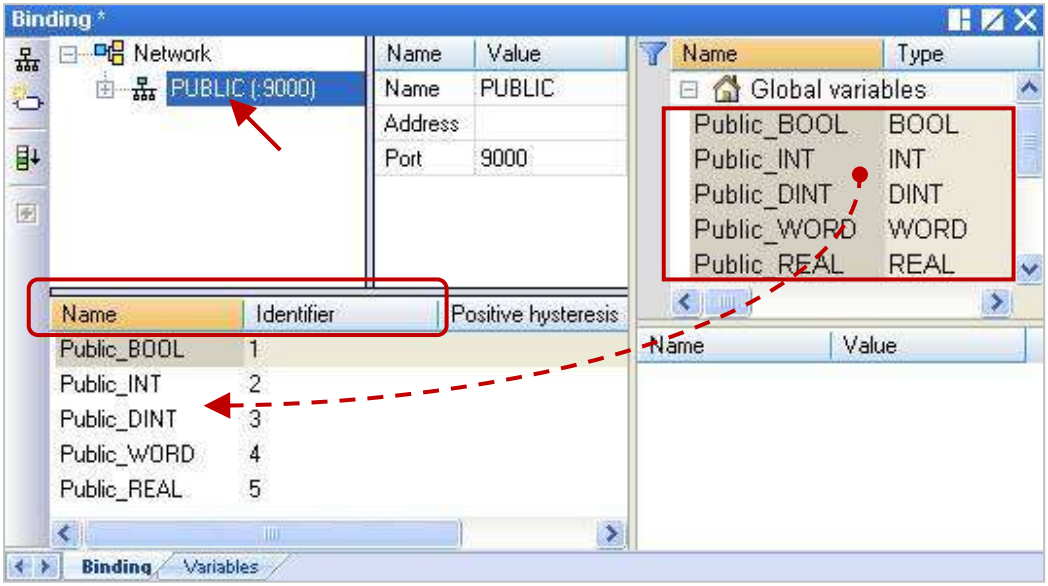

3. Before publishing these variables, make sure you have declared them in the Variables Area. Click “Global variables” and press the “Ins” key to insert a new variable. The following table demonstrates variables using in the “Test_3” project and you can declare them according to the needs of your application.

| Variable name | Type |
|---|---|
| Public_BOOL | BOOL |
| Public_INT | INT |
| Public_DINT | DINT |
| Public_WORD | WORD |
| Public_REAL | REAL |




4. As the figure below, click on "PUBLIC(:9000)" and drag all the needed variables to the "Name" area. The "Identifier" will generate an address number automatically. If any other VB or .NET program wants to use these public variables, it must set to the same address number (ID).

**NOTE:**
The "PUBLIC" allows to use up to 8192 variables, and the "Identifier" number JUST can be 1 to 8192.




The following procedure will show you how to use the "pub_string" function to publish the Win-GRAF String variable in the ST program.

**Syntax:**

```
Pub_string(Address, String_val) ;
```

Address: The public address number, and its range can be 1 to 1024
String_val: The name of String variable.

**Variables description:**

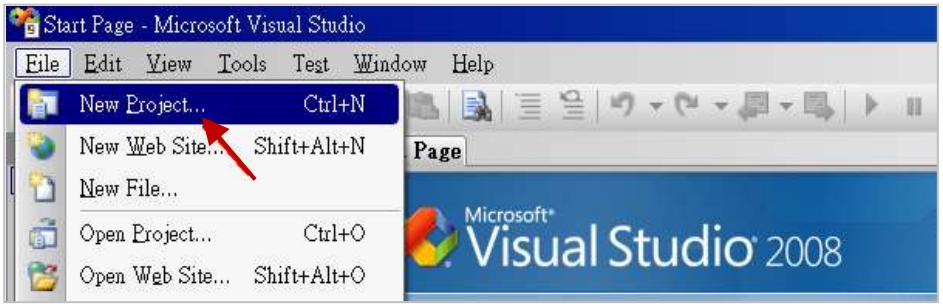
| Name | Type | Description |
|---|---|---|
| Init | BOOL | Set the initial value as TRUE. |
| Tmp_val | BOOL | TRUE: Binding succeeds.<br>FALSE: Binding fails. |
| msg1 | STRING, Length is 100 | String variable for demo purpose.<br>**NOTE: The String length could be 1 to 255.** |
| msg2 | STRING, Length is 32 | |
| msg3 | STRING, Length is 60 | |

**ST program:**

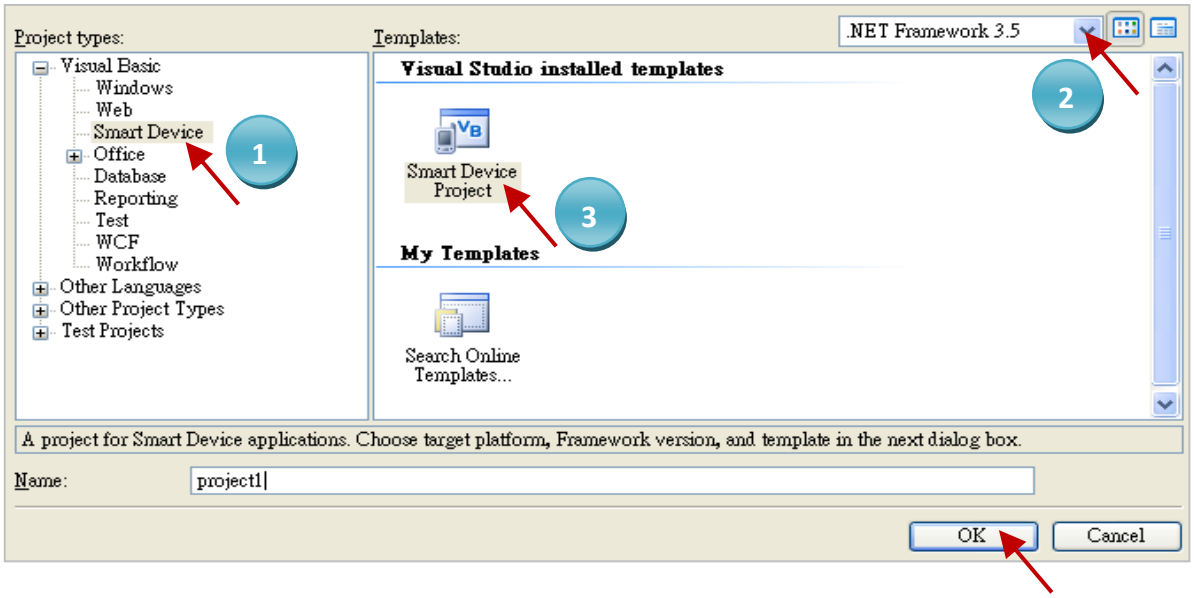
```
If  init  then
  Init := false;
(*add address 1 for share string val *)
  Tmp_val := pub_string(1,msg1);

(*add address 2 for share string val *)
  Tmp_val := pub_string(2,msg2);

(*add address 3 for share string val *)
  Tmp_val := pub_string(3,msg3);

End_if;
```

## 13.3 Create a new VB.NET project

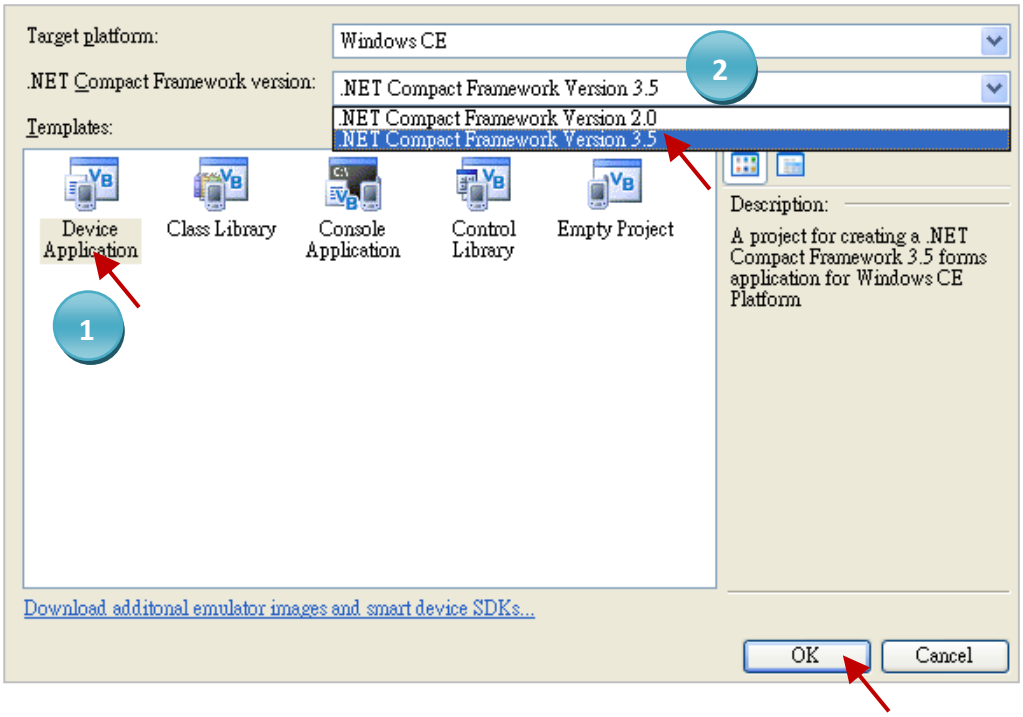
1. First, run Microsoft Visual Studio .NET 2008 software, and then choose “File” > “New Project”.




2. Click “Smart Device” on the left, and then select “.NET Framework 3.5” and “Smart Device Project”. Entering a proper project name and click “OK”.




3. Select the "Device Application" and "Windows CE" and “.NET Compact Framework Version 3.5”, then click “OK”.

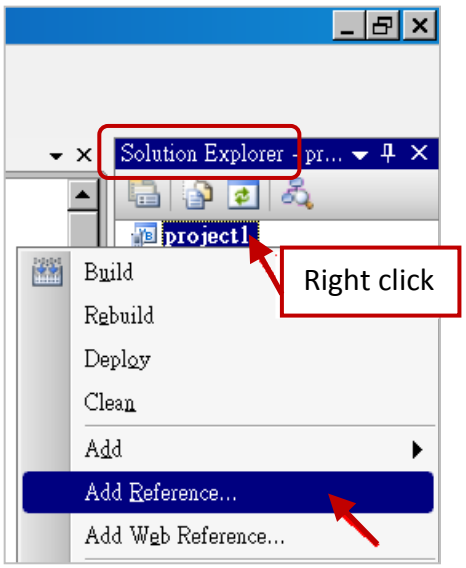


### 13.3.1 Add Project Reference

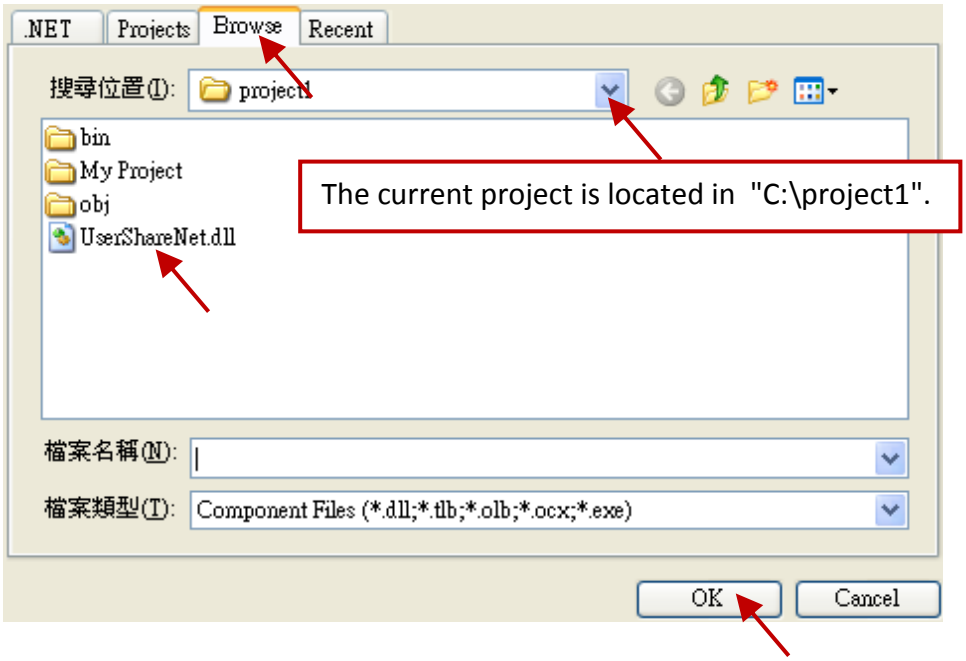
The “UserShareNet.dll” library contains all functions of data exchange with Win-GRAF variables. Before you use the “UserShareNet” keyword in the program, you must add the “UserShareNet.dll” into the reference list of your project.

1. Copy the “UserShareNet.DLL” from Win-PAC’s shipment CD (\napdos\Win-GRAF\WP-8xx8\ vb.net_2008_demo\wp_vb01\vb01\) to your project folder (e.g., “C:\project1\”)

2. Right click on the project name (e.g., “project1”) in the “Solution Explorer” window, and then select “Add Reference …”.

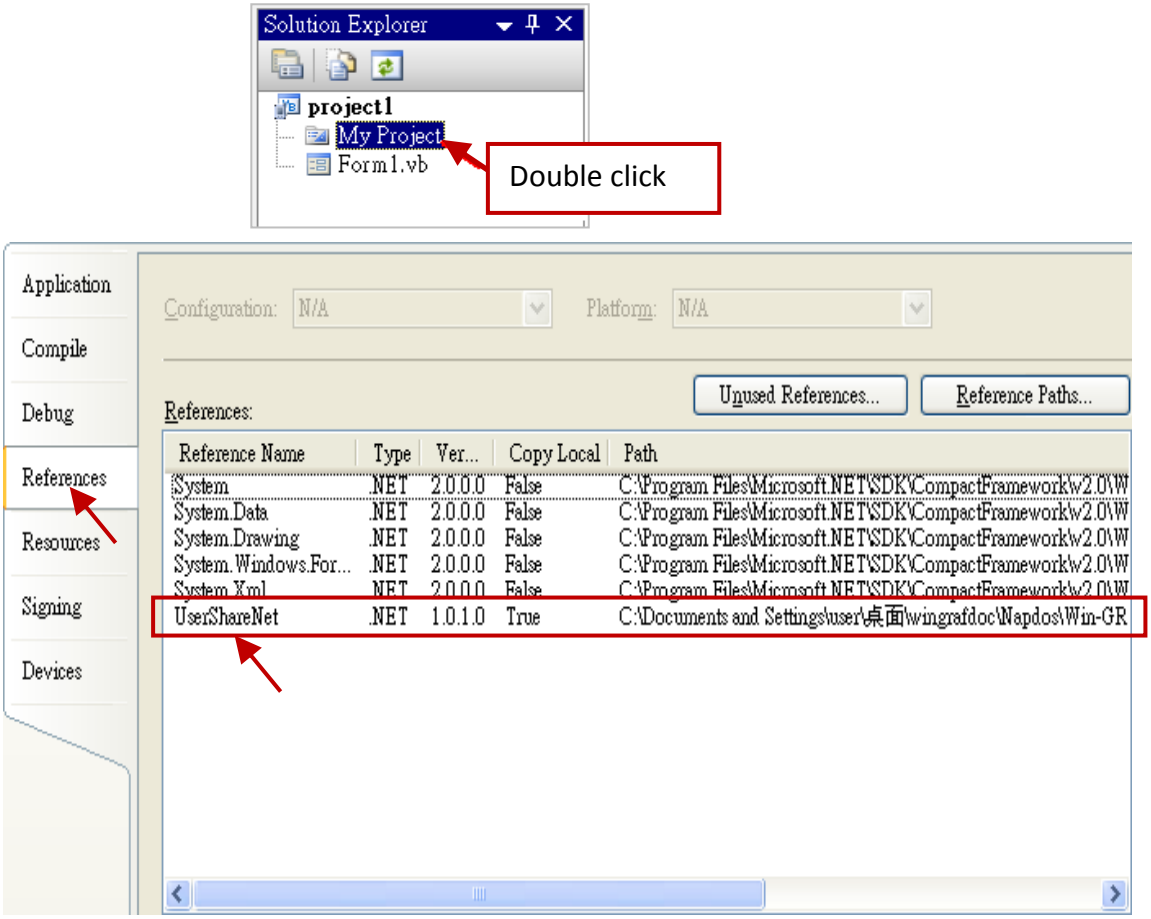



3. Click the “Browse” tab and select the “UserShareNet.dll” from your project location.

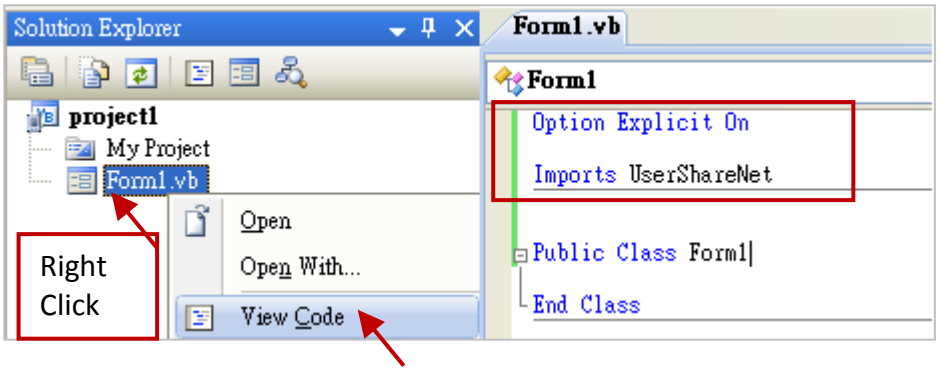



Note: You may copy the “UserShareNet.dll” from the CD-ROM to your current project path first. Then add it to the project reference.

4. When “UserShareNet.dll” is added, please double click on “My Project” to check if the “UserShareNet.dll” is well added.




5. Right-click on the “Form1.vb” and select “View Code” from the pop-up. Move cursor to top and insert the “Option Explicit On” and “Imports UserShareNet” in the first two statements.

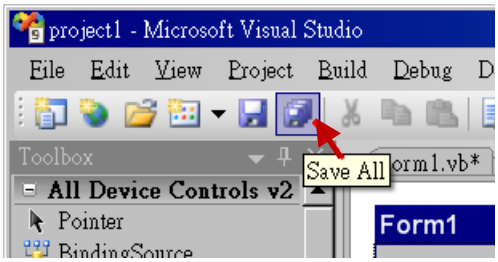



Then you can design all required objects and actions inside your VB Forms.
(Refer the Chapter 13.5 for more information about using functions in the “UserShareNet.dll”.)

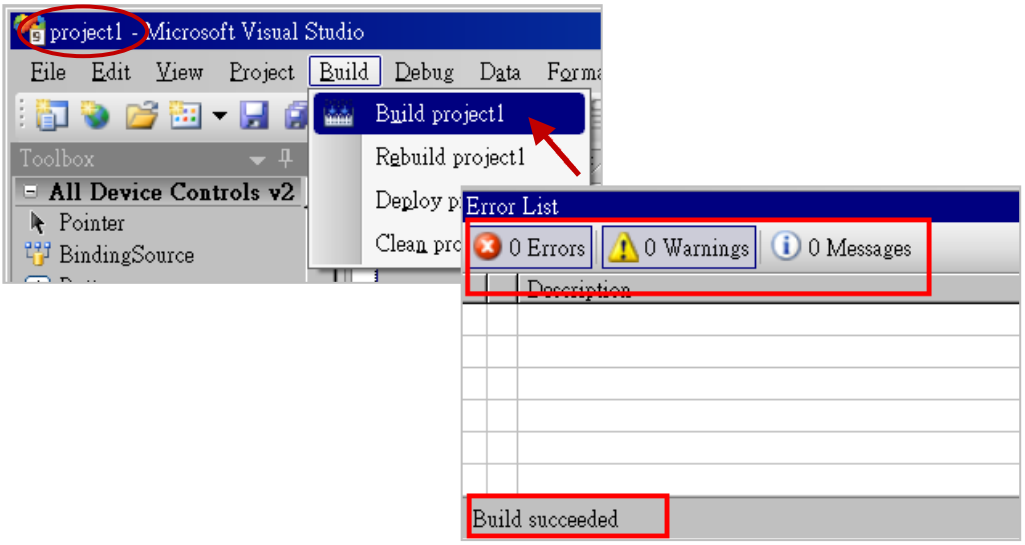
## 13.4 Compiling the Application

When you have finished writing a program, you can build (compile) an application by the following steps.

1. Remember to save at any time for safety.




2. Then compile (Build) the project. The result is listed in the "Error List" windows at the bottom.




3. You can find the execution file in

<Your VB.net Project folder> \bin\Release\ **<project_name>.exe**

Please copy this execution file to the WP-8xx8's \System_Disk\Win-GRAF\ path to run it.

**Note:**

**The user may copy the VB.net execution file to another path to run it, but there should contain at least two DLL files with it or it cannot run correctly.**

For ex, the project1.exe can run in the \Micro_SD\ folder if there are three files in it. The "project1.exe" , "UserShareNet.dll" and "Quicker.dll" . (The "UserShareNet.dll" and "Quicker.dll" can be copied from the Win-GRAF PAC's "\System_disk\Win-GRAF\" path)

## 13.5 UserShareNet.dll

This section we will focus on the description of the application example of UserShareNet.dll functions. There are some functions that can be used to read/write data from/to the Win-GRAF soft-logic. The functions of UserShareNet.dll can be divided into as listed below:

1. R/W Boolean
2. R/W 8-bit Integer
3. R/W 16-bit Integer
4. R/W 32-bit Integer
5. R/W 64-bit Integer
6. R/W 32-bit Float
7. R/W 64-bit Float
8. R/W String

※ **Refer the "Appendix A" to get familiar with the definition of Win-GRAF variables.**

### 13.5.1 R/W Boolean Functions

#### Set_BOOL

**Description:**

This function is to set a value to a Win-GRAF Boolean variable.

**Syntax:**

```
UserShare.Set_BOOL ( iUserAddress  As System.UInt16 , ByVal iStatus As byte) as Byte
```

**Parameter:**

iUserAddress : Address of the Variable (1 to 8192)

iStatus : Set the status. For instance, iStatus = 1 for True, iStatus = 0 for False

**Example:**

'Set the Win-GRAF BOOL variable with address 1 to True.

UserShare.Set_BOOL(Convert.ToUInt16(1), 1)

**Demo program:**

CD-ROM: \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb01

## Get_BOOL

**Description:**

This function is to get the value from a Win-GRAF BOOL variable.

**Syntax:**

**UserShare.Get_BOOL ( iUserAddress As System.UInt16 , ByRef iStatus As byte)**

**Parameter:**

iUserAddress : Address of the Variable (1 to 8192)

iStatus : Get the variable value , iStatus = 1 for True, iStatus = 0 for False

**Example:**

```
'Get the value of Win-GRAF BOOL variable with address 1.
Dim iStatus As Byte
UserShare.Get_BOOL(Convert.ToUInt16(1), iStatus)
```

**Demo Program:**

CD-ROM: \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb01

### 13.5.2 Integer R/W Functions

■ **Set_SINT** ■ **Set_INT** ■ **Set_DINT** ■ **Set_LINT**

**Description:**

These functions are to set 8-bit Integer, 16-bit Integer, 32-bit integer & 64-bit Integer value to Win-GRAF integer variables.

**Syntax:**

**UserShare.Set_SINT (ByVal *iUserAddress* As System.UInt16 , ByVal *iStatus* As SByte) As Byte**

**UserShare.Set_INT (ByVal *iUserAddress* As System.UInt16 , ByVal *iStatus* As Short) As Byte**

**UserShare.Set_DINT (ByVal *iUserAddress* As System.UInt16 , ByVal *iStatus* As Integer) As Byte**

**UserShare.Set_LINT (ByVal *iUserAddress* As System.UInt16 , ByVal *iStatus* As long) As Byte**

**Parameter:**

iUserAddress : Address of Variable. (1 to 8192)
iStatus : the value of 8-bit Integer, 16-bit Integer, 32-bit Integer or 64-bit Integer.

**Example:**

'Set a 32-bit integer value "1234567" to the Win-GRAF DINT variable with address "1".
UserShare.Set_DINT(Convert.ToUInt16(1), Convert.ToInt32(1234567) )

'Set a 16-bit integer value "-1234" to the Win-GRAF INT variable with address "2".
UserShare.Set_INT(Convert.ToUInt16(3), Convert.ToInt16(-1234) )

'Set a 64-bit Integer value "123456789012345" to the Win-GRAF LINT variable with address "3".
UserShare.Set_LINT(Convert.ToUInt16(3), Convert.ToInt64(123456789012345) )

'Set a 8-bit Integer value "125" to the Win-GRAF SINT variable with address "4".
UserShare.Set_SINT(Convert.ToUInt16(3), Convert.ToSByte(125) )

**Demo Program:**

CD-ROM:

1. \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb02 for R/W analog I/O
2. \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb03 for R/W internal long integer, Timer and Real (floating-point) values.

## ■ Get_SINT ■ Get_INT ■ Get_DINT ■ Get_LINT

**Description:**

These functions are to get 8-bit integer, 16-bit integer, 32-bit integer & 64-bit integer value from Win-GRAF integer variables.

**Syntax:**

**UserShare. Get_SINT (ByVal *iUserAddress* As System.UInt16 , ByRef *iStatus* As SByte) As Byte**

**UserShare. Get_INT (ByVal *iUserAddress* As System.UInt16 , ByRef *iStatus* As Short) As Byte**

**UserShare.Get_DINT (ByVal *iUserAddress* As System.UInt16 , ByRef *iStatus* As Integer) As Byte**

**UserShare. Get_LINT (ByVal *iUserAddress* As System.UInt16 , ByRef *iStatus* As long) As Byte**

**Parameter:**

iUserAddress : Address of Variable (1 to 8192)

iStatus : Get the 8-bit integer, 16-bit integer, 32bit-integer or 64-bit integer value.

**Example:**

```
Dim Dlong_val As Int64
Dim short_val As Int16
Dim long_val As Int32
Dim Sbyte_val as byte

'Get 64-bit integer value from the Win-GRAF LINT variable with address "7".
UserShare.Get_LINT(Convert.ToUInt16(7), Dlong_val)

'Get 32-bit integer value from the Win-GRAF DINT variable with address "8".
UserShare.Get_DINT(Convert.ToUInt16(8), long_val)

'Get 16-bit integer value from the Win-GRAF INT variable with address "9".
UserShare.Get_INT(Convert.ToUInt16(9), short_val)

'Get 8-bit integer value from the Win-GRAF SINT variable with address "10".
UserShare.Get_SINT(Convert.ToUInt16(9), sbyte_val)
```

**Demo program:**

CD-ROM:

1. R/W analog I/O :
   \napdos\Win-GRAF\demo-project\vb.net_2008_demo \demo_vb02
2. R/W internal long integer, Timer and Real (floating-point) values :
   \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb03

### 13.5.3 R/W Real Variable Functions

**■ Get_REAL ■ Get_LREAL**

**Description:**
These functions are to get 32-bit REAL and 64-bit double from the Win-GRAF REAL/LREAL variable.

**Syntax:**

**UserShare. Get_REAL (ByVal *iUserAddress* As System.UInt16 , ByRef *iStatus* As Single) As Byte**

**UserShare. Get_LREAL(ByVal *iUserAddress* As System.UInt16 , ByRef *iStatus* As Double) As Byte**

**Parameter:**
iUserAddress : Address of Variable (1 to 8192)
iStatus : Get the 32-bit REAL or 64-bit double value.

**Example:**

```
Dim float_val As Single
Dim double_val As Double

'Get 64-bit double value from the Win-GRAF LREAL variable with address "7".
UserShare.Get_LREAL(Convert.ToUInt16(7), double_val)

'Get 32-bit REAL value from the Win-GRAF REAL variable with address "8".
UserShare.Get_REAL(Convert.ToUInt16(8), float_val)
```

**Demo program:**
CD-ROM:
1. R/W analog I/O:
   \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb02
2. R/W internal long integer, Timer and Real (floating-point) values :
   \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb03

## ■ Set_REAL ■ Set_LREAL

**Description:**

These functions are to set 32-bit REAL and 64-bit double value to the Win-GRAF REAL/LREAL variable.

**Syntax:**

**UserShare. Set_REAL (ByVal *iUserAddress* As System.UInt16, ByVal *iStatus* As Single) As Byte**

**UserShare. Set_LREAL(ByVal *iUserAddress* As System.UInt16, ByVal *iStatus* As Double) As Byte**

**Parameter:**

iUserAddress : Address of Variable. (1 to 8192)
iStatus : Set the 32-bit REAL or 64-bit double.

**Example:**

'Set a 64-bit double value "11234.234567" to the Win-GRAF LREAL variable with address "1".
UserShare.Set_LREAL(Convert.ToUInt16(7),Convert.ToDouble(11234.234567))

'Set a 32-bit REAL value "123.12" to the Win-GRAF REAL variable with address "8".
UserShare.Set_REAL(Convert.ToUInt16(8), Convert.ToSingle (123.12))

**Demo program:**

CD-ROM:

1. R/W analog I/O : \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb02
2. R/W internal long integer, Timer and Real (floating-point) values : \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb03

### 13.5.4 R/W String Variable Functions

#### Get_STRING

**Description:**
This function is to get a Win-GRAF String variable.

**Syntax:**

**UserShare. Get_STRING (ByVal *iUserAddress* As System.UInt16, ByVal *msg()* As Byte) As Byte**

**Parameter:**
iUserAddress : Address of Variable (1 to 1024)
msg() : Get the string value.

**Example:**

```
Dim str_val As String
Dim msg() As Byte

‘Get String value of the Win-GRAF String variable with address “7”.
UserShare.Get_STRING(Convert.ToUInt16(7),msg )
str_val= byte_array_to_unicode(msg)

 Private Function byte_array_to_unicode(ByVal buf() As Byte) As String
      Dim tmpmsg As String
      If buf.Length > 255 Then
         Return Nothing
      End If
tmpmsg = System.Text.Encoding.GetEncoding("UTF-8").GetString(buf, 0,  buf.Length)

      Return tmpmsg
  End Function
```

**Demo program:**
CD-ROM:
1. R/W String variable:
   \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb04

## Set_STRING

**Description:**

This function is to set a String value to the Win-GRAF String variable.

**Syntax:**

**UserShare. Set_STRING (ByVal *iUserAddress* As System.UInt16, ByVal *msg()* As Byte) As Byte**

**Parameter:**

iUserAddress : Address of Variable. (1 to 1024)

msg() : the string value.

**Example:**

```
Dim str_val As String=”Hello World”
Dim msg() As Byte

msg= unicode_to_byte_array(str_val)

‘Set a string value “Hello World” to the Win-GRAF String variable with address “7”.
UserShare.Set_STRING(Convert.ToUInt16(7),msg )

‘Convert String to byte array.
Private Function unicode_to_byte_array(ByVal msg As String) As Byte()
    Dim tmpbuf() As Byte
    If msg.Length > 255 Then
       Return Nothing
    End If
    tmpbuf = System.Text.Encoding.GetEncoding("UTF-8").GetBytes(msg)
    Return tmpbuf
End_Function
```

**Demo program:**

CD-ROM:

1. R/W String variable:
   \napdos\Win-GRAF\demo-project\vb.net_2008_demo\demo_vb04

### 13.5.5 How to use VB.NET R/W to Win-GRAF String Variable

Before .NET program write to Win-GRAF String variable. The String-type has to convert to byte array. If you need to read Win-GRAF String variable. Then you have to convert byte array to String. There is a VB.NET example to show how to convert each other.

(Encode :UTF-8 )：

**Convert String to byte array**

```
  Private Function unicode_to_byte_array(ByVal msg As String) As Byte()
    Dim tmpbuf() As Byte
    If msg.Length > 255 Then
      Return Nothing
    End If

    tmpbuf = System.Text.Encoding.GetEncoding("UTF-8").GetBytes(msg)
    Return tmpbuf
  End Function
```

**Convert byte array to string**

```
  Private Function byte_array_to_unicode(ByVal buf() As Byte) As String
    Dim tmpmsg As String
    If buf.Length > 255 Then
      Return Nothing
    End If
tmpmsg = System.Text.Encoding.GetEncoding("UTF-8").GetString(buf, 0, buf.Length)

    Return tmpmsg
  End Function
```

# Chapter 14 C# .net 2008 Program Running in WP-8xx8 Access to Win-GRAF Variables

This chapter lists the procedure for creating the first demo program by Visual Studio .NET 2008 development tool. There are some sample programs in the WP-8xx8 CD-ROM.

**C# demo:**

CD-ROM : \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\

demo_CSharp01 : Digital I/O demo with one I-87055W in slot 0 of the WP-8xx8.

demo_CSharp02 : Analog I/O demo with one I-87024W in slot 1 and one I-8017HW in slot 2.

demo_CSharp03 : Read / Write Win-GRAF internal integers, timers and real variables. (No I/O)

demo_CSharp04 : Read/Write Win-GRAF internal String variables. (No I/O)

**Win-GRAF demo:**

CD-ROM : \napdos\Win-GRAF\demo-project\

"demo_vb01.zip", "demo_vb02.zip", "demo_vb03.zip", "demo_vb04.zip"

## 14.1 Add an Existing Win-GRAF Project from a ZIP

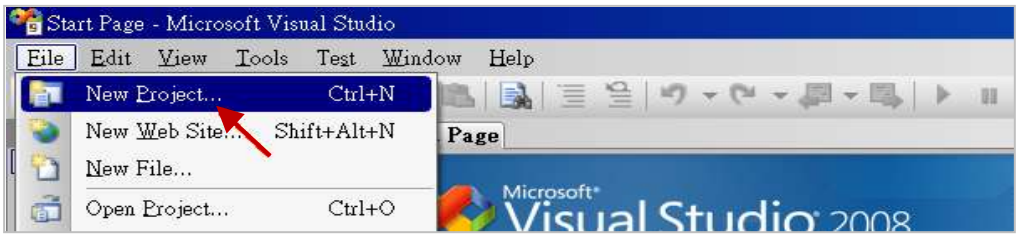
Please refer the Chapter 13.1

## 14.2 Publishing the Win-GRAF Variable for .NET

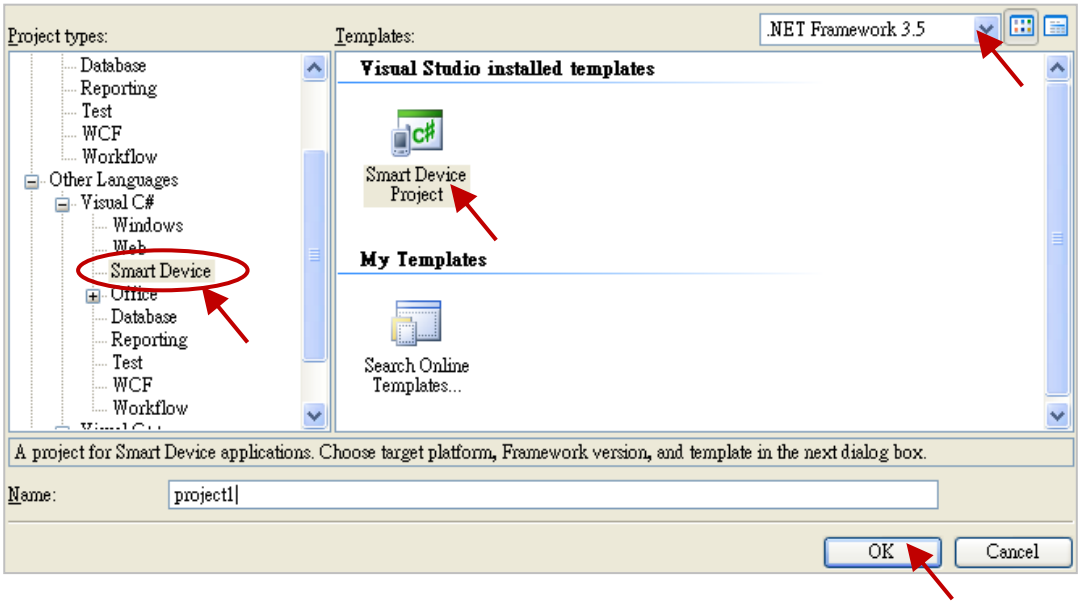
Please refer the Chapter 13.2

## 14.3 Create a New C# Project

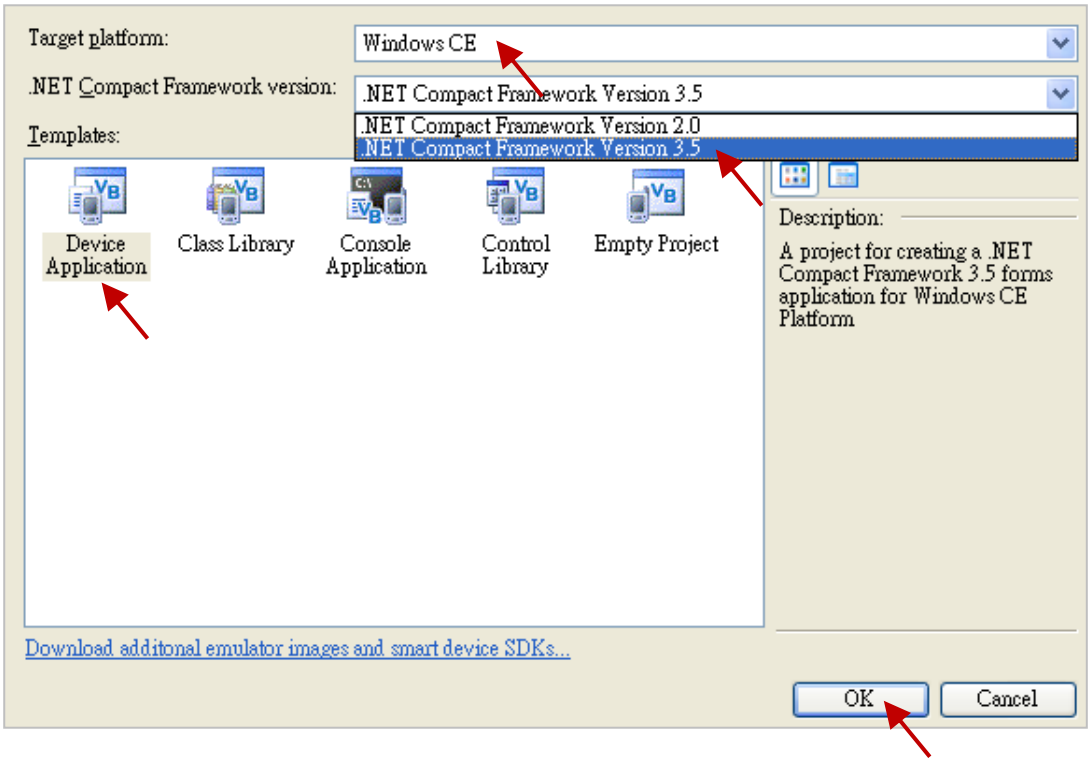
1. First, users need to open Microsoft Visual Studio .NET 2008 software. And then in the menu of “File”, please run the “New Project” .



2. Check the “Smart Device” on the left, then selecting the “.NET framework 3.5” and “Smart Device Project”. Then entering a proper project name and the last click on “OK”.




3. Select the "**Device Application**" and "**Windows CE**" and “**.NET Compact Framework Version 3.5**”, then click on “OK”.

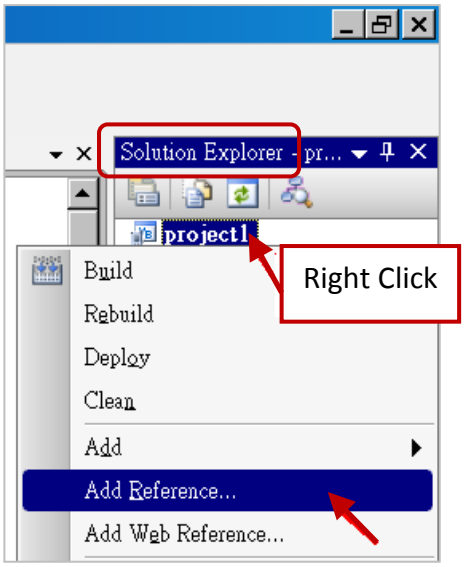


### 14.3.1 Add C# Project Reference

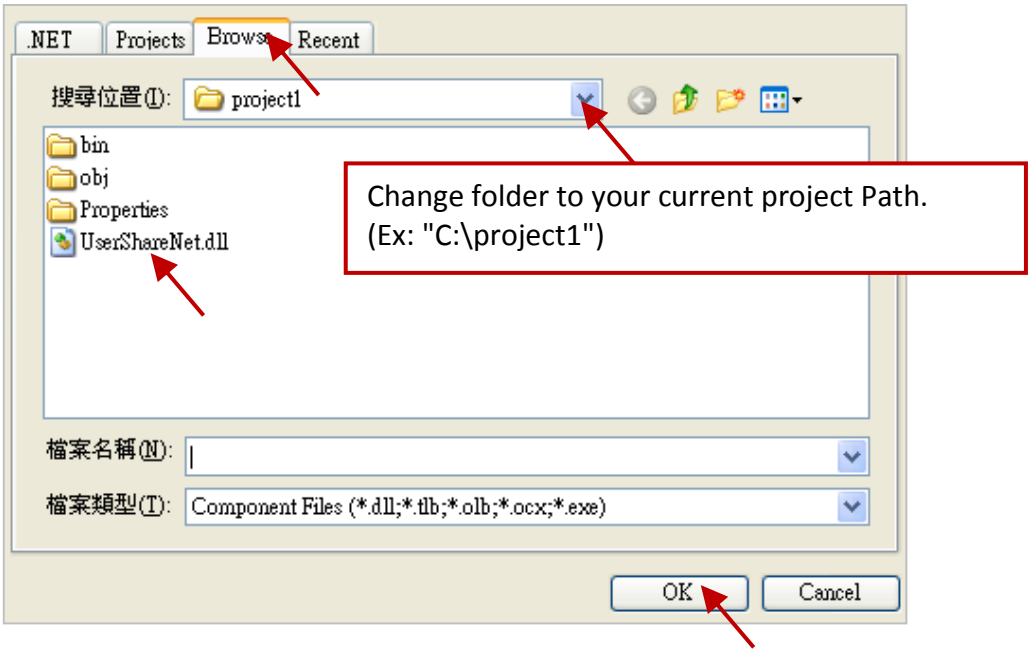
The “UserShareNet” library contains all modules’ functions. Before you use the “UserShare” keyword in the program, you must add the “UserShareNet.dll” into the reference list of your application.

1. Copy the “UserShareNet.DLL” from WP-8xx8 CD-ROM: **\napdos\Win-GRAF\WP-8xx8\CSharp.net_2008_demo\demo_CSharp01** to your project folder(ex: C:\project1\)

2. Right click on the Project name on the right hand side , then select “Add Reference …”

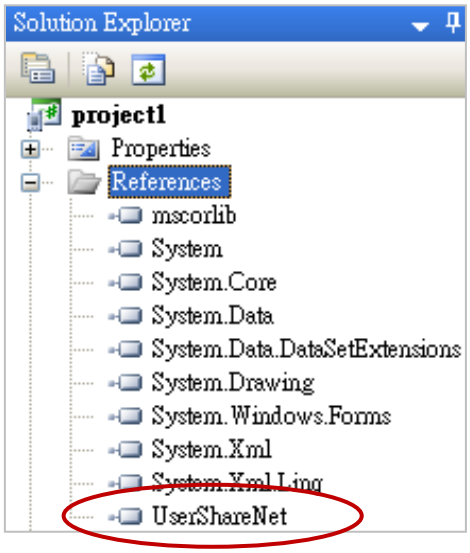



3. Click the “**Browse**” button. Select the “UserShareNet.dll” from your project location.

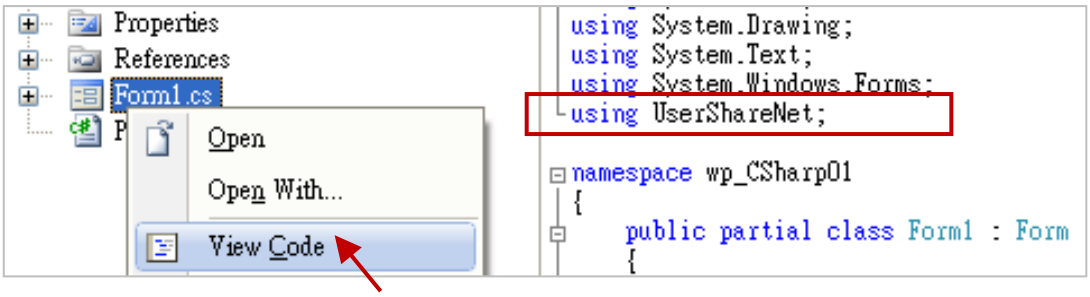



**Note:** **You may copy the “UserShareNet.dll” from the CD-ROM to your current project path first. Then add it to the project reference.**

4. When “UserShareNet.dll” are added, you can see them in the solution explorer as below.




5. Right-click on the “**Form1.cs**” and select “**View Code**” from the pop-up. Move cursor to top and insert the “**using UserShareNet;**” in the first statements.

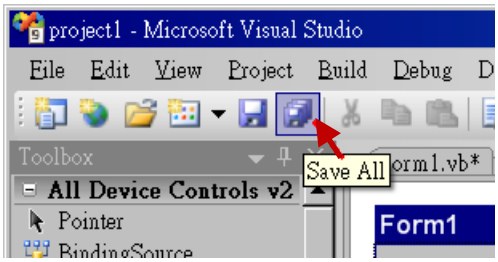



Then you can design all required objects and actions inside your C# Forms.
(Refer the Section 14.5 for more information about using functions in the “UserShareNet.dll”.)

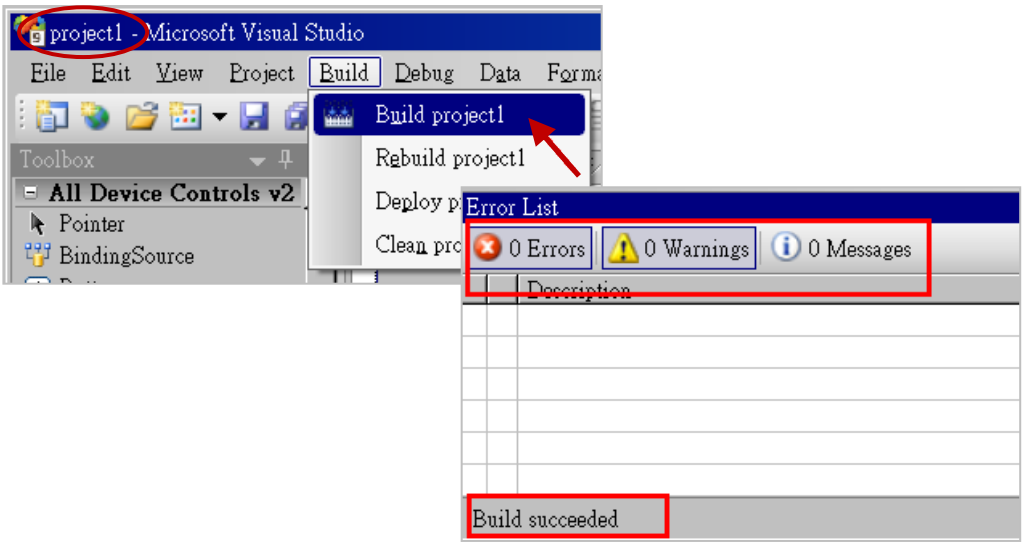
## 14.4 Compiling the Application Program

When you have finished writing a program, you can build (compile) an application by the following steps.

1. Remember to save at any time for safety.




2. Then compile (Build) the project . The result is listed in the “Error List” windows at the bottom.




3. You can find the execution file in

<Your C# .net Project folder> \bin\Release\ <project_name>.exe

Please copy this execution file to the WP-8xx8’s \System_Disk\Win-GRAF\ path to run it.

**Note:**

The user may copy the C#.net execution file to another path to run it, but there should contain at least two DLL files with it or it cannot run correctly. For ex, the project1.exe can run in the \Micro_SD\ path if there are three files in it. The “project1.exe”, “UserShareNet.dll” and, “Quicker.dll” .
(The “UserShareNet.dll” and “Quicker.dll” can be copied from the Win-GRAF PAC’s “\System_disk\Win-GRAF\” path)

## 14.5 UserShareNet.DLL

This section we will focus on the description of the application example of UserShareNet.DLL functions. There are some functions that can be used to read/write data from/to the Win-GRAF variable. The functions of UserShareNet.DLL can be divided into as listed below

1. R/W Boolean
2. R/W 8-bit Integer
3. R/W 16-bit Integer
4. R/W 32-bit Integer
5. R/W 64-bit Integer
6. R/W 32-bit Float
7. R/W 64-bit Float
8. R/W 32-bit String

**※ Refer the "Appendix A" to get familiar with the definition of Win-GRAF variables.**

### 14.5.1 R/W Boolean Functions

#### ■ Set_BOOL

**Description:**
This function is to set a value to a Win-GRAF Boolean variable.

**Syntax:**

```
UserShare.Set_BOOL(ushort iUserAddress, byte iStatus)
```

**Parameter:**
iUserAddress : Address of Variable (1 to 8192)
iStatus : Set the status. For instance, iStatus = 1 for True, iStatus = 0 for False

**Example:**

```
// Set the Win-GRAF BOOL variable with address 1 to True.
 UserShare.Set_BOOL(Convert.ToUInt16(1), 1);
```

**Demo program:**
CD-ROM : \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp01

## Get_BOOL

**Description:**

This function is to get the value from a Win-GRAF BOOL variable.

**Syntax:**

```
UserShare.Get_BOOL(ushort iUserAddress, out byte iStatus)
```

**Parameter:**

iUserAddress : Address of Variable. (1 to 8191)

iStatus : Get the variable status , iStatus = 1 for True, iStatus = 0 for False.

**Example:**

```
Byte iStatus=0;
// Get the value of Win-GRAF BOOL variable with address 1.
UserShare.Get_BOOL(Convert.ToUInt16(1),out iStatus);
```

**Demo program:**

CD-ROM: \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp01

### 14.5.2 R/W Integer Functions

■ Set_SINT ■ Set_INT ■ Set_DINT ■ Set_LINT

**Description:**
These functions are to set 8-bit Integer, 16-bit Integer, 32-bit integer & 64-bit Integer value to Win-GRAF integer variables.

**Syntax:**

```
UserShare.Set_SINT(ushort iUserAddress , sbyte iStatus)

UserShare.Set_INT(ushort iUserAddress , short iStatus)

UserShare.Set_DINT(ushort iUserAddress, int iStatus)

UserShare.Set_LINT(ushort iUserAddress, long iStatus)
```

**Parameter:**
iUserAddress : Address of Variable. (1 to 8192)
iStatus : Set the 8-bit Integer, 16-bit Integer, 32-bit Integer or 64-bit Integer.

**Example:**

```
// Set a 32-bit integer value “1234567” to the Win-GRAF DINT variable with address “1”.
int temp1=1234567;
UserShare.Set_DINT(Convert.ToUInt16(1), temp );

// Set a 16-bit integer value “-1234” to the Win-GRAF INT variable with address “2”.
short temp2= -1234;
UserShare.Set_INT(Convert.ToUInt16(2), temp2 );

// Set a 64-bit Integer value “123456789012345” to the Win-GRAF LINT variable with address “3”.
long temp3=123456789012345;
UserShare.Set_LINT(Convert.ToUInt16(3),  temp3 );

// Set a 8-bit Integer value “125” to the Win-GRAF SINT variable with address “4”.
Sbyte temp4=125;
UserShare.Set_SINT(Convert.ToUInt16(4),  temp4 );
```

**Demo program:**
CD-ROM:

1. R/W analog I/O:
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp02
2. R/W internal Boolean ,long integer, Timer and Real (floating-point) values :
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp03

## ■ Get_SINT ■ Get_INT ■ Get_DINT ■ Get_LINT

**Description:**

These functions are to get 8-bit integer, 16-bit integer, 32-bit integer & 64-bit integer value from Win-GRAF integer variables.

**Syntax:**

**UserShare.Get_SINT(ushort iUserAddress, out sbyte iStatus)**

**UserShare.Get_INT(ushort iUserAddress, out short iStatus)**

**UserShare.Get_DINT(ushort iUserAddress, out int iStatus)**

**UserShare.Get_LINT(ushort iUserAddress, out long iStatus)**

**Parameter:**

iUserAddress : Address of Variable (1 to 8192)
iStatus : Get the value of Win-GRAF integer variables.

**Example:**

```
Int64 Dlong_val;
Int16 short_val;
Int32 long_val ;
sbyte sbyte_val;

// Get 64-bit integer value from the Win-GRAF LINT variable with address “7”.
UserShare.Get_LINT(Convert.ToUInt16(7),out Dlong_val);

// Get 32-bit integer value from the Win-GRAF DINT variable with address “8”.
UserShare.Get_DINT(Convert.ToUInt16(8),out long_val);

// Get 16-bit integer value from the Win-GRAF INT variable with address “9”.
UserShare.Get_INT(Convert.ToUInt16(9),out short_val);

// Get 8-bit integer value from the Win-GRAF SINT variable with address “10”.
UserShare.Get_SINT(Convert.ToUInt16(9),out sbyte_val)
```

**Demo program:**

CD-ROM:

1. R/W analog I/O:
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp02
2. R/W internal Boolean, long integer, Timer and Real (floating-point) values:
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp03

### 14.5.3 R/W Real variable Functions

■ **Get_REAL** ■ **Get_LREAL**

**Description:**

These functions are to get 32-bit REAL and 64-bit double from the Win-GRAF.

**Syntax:**

**UserShare. Get_REAL (System.UInt16 *iUserAddress*, out float *iStatus*)**

**UserShare. Get_LREAL(ByVal *iUserAddress* As System.UInt16 , out Double *iStatus*)**

**Parameter:**

iUserAddress : Address of Variable (1 to 8192)
iStatus : Get the 32-bit REAL or 64-bit double value.

**Example:**

```
float float_val;
double double_val;

// Get 64-bit double value from the Win-GRAF LREAL variable with address “7”.
UserShare.Get_LREAL(Convert.ToUInt16(7),out double_val);

// Get 32-bit REAL value from the Win-GRAF REAL variable with address “8”.
UserShare.Get_REAL(Convert.ToUInt16(8),out float_val);
```

**Demo program:**

CD-ROM:

1. R/W analog I/O:
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp02
2. R/W internal long integer, Timer and Real (floating-point) values :
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_ demo_CSharp03

## ■ Set_REAL ■ Set_LREAL

**Description:**

These functions are to set 32-bit REAL and 64-bit double value to the Win-GRAF REAL/LREAL variable.

**Syntax:**

**UserShare. Set_REAL ( ushort *iUserAddress* , float *iStatus* )**

**UserShare. Set_LREAL( ushort *iUserAddress* , Double *iStatus*)**

**Parameter:**

iUserAddress : Address of Variable. (1 to 8192)

iStatus : Set the 32-bit REAL or 64-bit double.

**Example:**

```
// Set a 64-bit double value “11234.234567” to the Win-GRAF LREAL variable with address “7”
UserShare.Set_LREAL(Convert.ToUInt16(7),Convert.ToDouble(11234.234567));

// Set a 32-bit REAL value “123.12” to the Win-GRAF REAL variable with address “2”.
UserShare.Set_REAL(Convert.ToUInt16(8), Convert.ToSingle (123.12));
```

**Demo program :**

CD-ROM:

1. R/W analog I/O:
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp02
2. R/W internal long integer, Timer and Real (floating-point) values :
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp03

### 14.5.4 R/W String variable Functions

#### Set_STRING

**Description:**
This function is to get a Win-GRAF String variable.

**Syntax:**

```
UserShare.Set_STRING (ushort  addr , Byte []  msg)
```

**Parameter:**
addr : Address of Variable (1 to 1024)
msg[] : Get the string value.

**Example:**

```
String str_val;
Byte[] msg;

// Get the String value of the Win-GRAF String variable with address “7”.
msg= unicode_to_byte_array(str_val);
UserShare.Set_STRING(Convert.ToUInt16(7),msg );

//Convert String to byte array.
private byte[] unicode_to_byte_array(string msg)
    {
      byte[] tmpbuf;
      if (msg.Length > 255)
        return null;

      tmpbuf = Encoding.GetEncoding("UTF-8").GetBytes(msg);
      return tmpbuf;
    }
```

**Demo program:**
CD-ROM:
1. R/W String variable :
   \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp04

## Get_STRING

**Description:**

This function is to set a String value to the Win-GRAF String variable.

**Syntax:**

| UserShare.Set_STRING (ushort addr , Byte [] msg) |
|---|

**Parameter:**

addr : Address of Variable. (1 to 1024)

msg[] : Set the string value.

**Example:**

```
String str_val= “Hello World”;
Byte[] msg;

// Set a string value “Hello World” to the Win-GRAF String variable with address “7”.
UserShare.Get_STRING(Convert.ToUInt16(7),msg );
str_val= byte_array_to_unicode(msg);

//Convert byte array to String
private string byte_array_to_unicode(byte[] buf)
 {
        string tmpmsg;
        if (buf.Length > 255)
           return null;

        tmpmsg = Encoding.GetEncoding("UTF-8").GetString(buf, 0, buf.Length);
        return tmpmsg;
  }
```

**Demo program:**

CD-ROM:

1. R/W String variable : \napdos\Win-GRAF\demo-project\CSharp.net_2008_demo\demo_CSharp04

### 14.5.5 How to Use C# to Convert Win-GRAF String Variable

Before .NET program write to Win-GRAF String variable. The String-type has to convert to byte array. (According your .NET program Encode. Ex: UTF-8) If you need to read Win-GRAF String variable. Then you have to convert byte array to String. There is an C# example to show how to convert each other.

**Example (Encode is UTF-8)**:

**//Convert String to byte array**

```
private byte[] unicode_to_byte_array(string msg)
    {
       byte[] tmpbuf;
       if (msg.Length > 255)
          return null;

       tmpbuf = Encoding.GetEncoding("UTF-8").GetBytes(msg);
       return tmpbuf;
    }
```

**//byte array to string**

```
private string byte_array_to_unicode(byte[] buf)
 {
       string tmpmsg;
       if (buf.Length > 255)
          return null;

       tmpmsg = Encoding.GetEncoding("UTF-8").GetString(buf, 0, buf.Length);
       return tmpmsg;
  }
```